Report of the International Grid Performance Workshop 2005

Steering Committee:

Chair: Jennifer M. Schopf, Argonne National Laboratory and NeSC
Bill Gropp, Argonne National Laboratory
Stephen Jarvis, University of Warwick
Mark Leese, Daresbury Laboratory
Brian Tierney, Lawrence Berkeley National Laboratory

Advisory Committee:

Mark Baker, University of Portsmouth
Fran Berman, UCSD
Jack Dongarra, University of Tennessee
Ian Foster, Argonne National Laboratory
Bill Gropp, Argonne National Laboratory
Tony Hey, UK e-Science Core Programme

Date: **7-2005**

Abstract:

This document reports on the third International Grid Performance Workshop, held at the National e-Science Centre in Edinburgh on 22–23 June 2005. The meeting focused on the performance needs of three example applications: RealityGrid, FusionGrid, and LCG. To complement the application use cases, a number of state-of-the-art Grid performance tools and techniques were also outlined, with the aim of identifying how these (or similar) tools might help the application scientists understand and address their performance needs. Analysis of the contributions to the workshop allows the current status of this research area to be determined, and a number of recommendations to be made.

Reference: Argonne National Laboratory MCS Technical Memorandum ANL/MCS-TM-288, and NeSC report UKeS-2005-07 available from http://www.nesc.ac.uk/technical_papers/UKeS-2005-07.pdf

Executive Summary

The International Grid Performance Workshop 2005 is the third in a set of workshops aimed at addressing the science of performance and the Grid. The workshop provided a forum for identifying and discussing the latest research and current application needs for performance and monitoring data for Grid systems. The meeting focused on the performance needs of three example applications, treating them as use cases. A number of state-of-the-art Grid performance tools and techniques were also presented, with the aim of identifying how these tools might help the application scientists understand and address their performance needs.

The conclusions from the meeting were as follows:

- ***A mismatch exists between application scientists and tool developers.*** Application developers often want simple tools for basic problems that work reliably, while tool builders are often funded to supply complex solutions to novel problems instead.
- ***Short-term and long-term needs vary strongly.*** Tool research often focuses on high-level services such as resource brokering, replica location, and metacatalogues, which—while clearly necessary—remain secondary requirements for applications.
- ***Performance simply is not on the critical path for many application projects.*** Applications that struggle to get code to execute correctly simply do not consider whether they are using resources efficiently or achieving good performance.
- ***Reliable solutions are clearly needed.*** In many cases, application scientists would prefer a reliable solution to one that might achieve the highest performance but fail.
- ***Clearer analysis of user requirements is needed.*** The needs of application scientists and tool builders needs to be clarified.
- ***Additional two-way communication is needed.*** Not only do tool builders need to know more about application needs, but also applications should be informed of current tools and their uses.

The following actions were suggested:

1. ***Survey application requirements.*** Paper surveys or interviews are needed to better understand requirements
2. ***Catalogue available tools.*** A catalogue of available performance tools should be compiled. This should encapsulate existing catalogues and would be made much more useful if feedback on the entries were enabled.
3. ***Identify key areas in Grid performance research.*** Short-term and long-term goals of this community should be defined, and these recommendations should be made available to the appropriate funding bodies.
4. ***Encourage joint application/tool developer projects.*** Efforts should be made to promote joint projects in order for either community to reap the benefits of the other.
5. ***Identify funding.*** Discussions should be held with funding bodies to determine how best to support multicountry and application scientists / tool developer projects.
6. ***Expand community outreach.*** Further discussion with, and between, tool and application scientists is needed.

Discussions were held as to whether an IGPW meeting should be arranged for next year. The feeling of the meeting was that IGPW is a productive forum and that in order to preserve continuity it would be good to have a meeting in 2006. A poll of previous attendees will be taken in six months time to ensure that this is still the general feeling. The theme of the next workshop will be that of *integration* between application scientists and tool developers. More application scientists will be encouraged to attend the workshop and paired talks will be arranged highlighting case studies where tool developers have worked alongside application scientists to meet specific performance needs. Demonstrations of tools and/or applications will be strongly encouraged.

1 Purpose of the Meeting

The International Grid Performance Workshop 2005 is the third in a set of workshops aimed at addressing the science of performance and the Grid. The workshop provided a forum for identifying and discussing the latest research and current application needs for performance and monitoring data for Grid systems. The 2005 meeting took place on June 22–23 at the National e-Science Centre in Edinburgh, UK.

Following the success of last year's workshop, this two-day working meeting focused on the performance needs of three example applications, treating them as use cases. A number of state-of-the-art Grid performance tools and techniques were also presented, with the aim of identifying how these (or similar) tools might help the application scientists understand and address their performance needs.

Applications are slowly being adapted to run over multiple administrative domains in a coordinated way, but they rarely achieve even a fraction of the possible performance of the underlying systems. In part this is because users rarely know what performance they could achieve. No current data or infrastructure exists for estimating baseline performance with which to determine the difference between how an application is currently running and what is possible with some tuning.

The following topics were addressed at the meeting:

- What performance criteria are most important to users?
- How are the resources monitored (i.e., how do the users know when the Grid is up)? How are failures detected (from job submission to file transfers)?
- How do users currently manage failures?
- What support tools are available for managing the performance of a Grid application, including off-the-shelf tools and application-specific tools. What additional tools are needed?
- What role do simulation and modelling play in a Grid environment, and how can we verify its appropriateness?
- How should reproducibility, scalability, and usability be ensured?

Section 2 summarises the contributions of application scientists from the RealityGrid, FusionGrid, and LCG projects, with particular reference to their performance requirements, the solutions that they employ, and the performance tools that are required to support their work. Section 3 provides an overview of the performance tools and techniques that were presented at the meeting and discusses the role of simulation and modelling in this work area. Following from these, Section 4 provides a "health check" on the current state of play, that is, how well the current work (and aspirations) of application scientists and tool developers is aligned. Section 5 documents the recommendations arising from the workshop for advancing this area of research and development. Section 6 briefly discusses plans for the next workshop.

Additional information on the workshop can be found in the appendices, including the agenda, participant list, and talk abstracts. Further details of the 2005 meeting can be found at http://www.mcs.anl.gov/~jms/GPW2005/. Similar material exists for the 2004 meeting, held at University College London: http://www.mcs.anl.gov/~jms/GPW2004/, which includes a link to the final report from that workshop. The work presented at the first International Grid Performance Workshop 2003 can be found in a special issue of *Concurrency and Computation: Practice and Experience*, Vol. 17, no. 2–4, 2005.

2 Summary of Contributions from the Application Scientists

Since the goals of the workshop were fairly well defined, all application speakers were requested to follow a tailored outline for their talk in order for the three use cases to be compared. This request included the following details:

- What are the specific performance criteria for the application?
- How are the hardware resources/services monitored?
- How do you detect and manage failures?
- What support tools do you use to manage the performance of your application, both in terms of off-the-shelf and "home grown"?
- What tool do you wish you had?

2.1 RealityGrid

The focus of RealityGrid is on high-performance computing and on employing Grid technology to aid computational studies of condensed matter—for example, oil, water, and surfactant mixtures; macromolecules (i.e., proteins); and semi-conductor surfaces. Steering and visualisation play an important role as they provide the user with increased control over their computation.

Highlighted performance criteria for RealityGrid and its supporting applications include the following:

- ***Will my job run*** – Is there sufficient memory for the input data, computation and expected results? Are there a sufficient number of processors on which to run the job? Is there sufficient maximum wall-clock time to complete the execution?
- ***How soon will my job start executing*** – What is the current queue length? What is the capacity of the machine on which the job is being run? How much checkpoint data is needed? What is the network connectivity between checkpoint host and compute machine?
- ***How long will my job take*** – How many processors are available, and what type of processor and interconnect are being used?
- ***Can I steer the job*** – Is it possible to improve performance by allowing a human to monitor the job? Can the job be migrated, and if so, what is the network connectivity between the source and destination sites? Will the job run at a convenient time (i.e., when will it be scheduled)?
- ***Will on-line visualisation be possible*** – this will allow for more detailed monitoring, although an addition performance constraint is the network connectivity and location of the associated visualisation engine.

Many of the performance criteria in RealityGrid depend on the application being run, the configuration of the queueing systems and the time the jobs need to get through the queues, and how important steering and visualisation are. This said, RealityGrid is characterised by the fact that most applications using high-performance computing (HPC) resources require that the application has been installed, and it is up to the user to check on resource status (through `gsissh` and `gstat`, for example) and the suitability of resources for a particular task.

In general, system simulations carry the highest cost (governed by the compute time on the high-performance resources). Therefore, standard HPC tools are used to verify performance and scalability. Failure management and steering rely heavily on checkpointing and migration; this situation in itself has performance implications (including delays for SOAP exchanges). Visualisation introduces additional costs, although a number of performance optimisations have been applied to alleviate these effects (certain data is collected only when

visualisation is active, the visualisation software reports the bandwidth it sees, GridFTP flags are tuned for maximum performance, etc.).

In the short term, the performance tools that the RealityGrid scientists would like to see developed include a Grid `top` or a Grid `stat` in order to obtain a quick graphical snapshot of the system, with the added ability to drill-down to specific machines.

Additional information on the RealityGrid can be found at: http://www.realitygrid.org/.

2.2 FusionGrid

FusionGrid is a collaborative project between the U.S. Department of Energy, the U.S. National Fusion facility, and a number of scientific laboratories (MIT, Princeton Plasma Physics Laboratory, Lawrence Berkeley National Laboratory, Argonne National Laboratory, Utah School of Computing and Princeton Department of Computer Science). The aim of the FusionGrid is to support rapid data analysis for fusion science experiments; the dominant performance metrics in these experiments are ***data analysis throughput*** and ***reliability***.

The fusion devices, called tokamaks, operate in pulses, or *shots*. Raw data is analysed in the twenty-minute intervals between these shots, and the results of this analysis are fed into the next shot. The computation between shots is highly organised, since each shot costs approximately $1 million. The purpose of FusionGrid is to make more efficient use of resources, share resources between sites, and develop a common set of software tools for fusion experiments.

Current performance tools include `MDSplus` (no relation to the Globus Toolkit MDS), and the Fusion Grid Monitor (`FGM`), which is tailor-made for general Grid-wide monitoring adapted from the U.S. national fusion facilities Data Access Monitor (`DAM`). Viewing the log files from this monitor not only is useful for dealing with current problems but also is important for locating and debugging common failures. To be useful to this project, a performance tool would need to ***monitor network QoS, CPU, mass storage, status of analyses and data consistency***; be ***easy enough for the application scientists to use***; be ***easy enough for developers to use*** (that is, be open source, easily extendible, and contain good APIs etc); and be ***reliable and dependable***. The financial implications of the experiments clearly influence the way in which performance is (and will be) investigated. It will not viable therefore to experiment with performance in a conventional sense – that is, record performance, make changes, re-run etc., as the financial penalties of a misdiagnosis would simply be too high.

Details on the FusionGrid can be found at http://www.fusiongrid.org/.

2.3 LCG2

The Large Hadron Collider (LHC) computing Grid (LCG) supports the world's largest superconducting accelerator at CERN, Geneva, Switzerland. Four detectors have been constructed in the 27 km accelerator and are operated by international collaborations of thousands of physicists, engineers and technicians. Data accumulated from the experiments that the LHC supports are predicted to accumulate at a rate of around 15 petabytes per year. This data will be analysed on LCG, and this process in itself will generate similar volumes of data. The middleware for LCG is being developed through the EGEE (Enabling Grids for E-sciencE) project; deployment and monitoring are also done jointly with EGEE.

Monitoring currently consists of ***identifying the state of each of the participating sites*** (of which there are currently 138 sites in 36 different countries); ***identifying what resources are***

currently being used; ***accounting***, in terms of how many resources are being used by different virtual organisations; and EGEE ***QoS monitoring***. The results of this analysis can be viewed through several tools (including `GridIce`). Accounting is facilitated through logs and is published through RGMA. ***Quality assurance*** is important, and the job success rate in the LCG is also monitored.

The performance requirements for fast, bulk-read access to databases were also discussed. Performance experiments based on `LFC` and `FiReMan`, both second-generation file access catalogues developed by CERN, were described.

Further detail on the LCG project can be found at http://lcg.web.cern.ch/LCG/. Additional information on the logging and bookkeeping and on information and monitoring subsystems of the gLite middleware (that supports EGEE) can be found at http://glite.web.cern.ch/glite/.

3 Summary of Contributions from the Performance Tool Developers

The tool developer talks were organised into four categories: tools to address whether the Grid is up, tools to understand job/file transfer failures, tools to help predict file transfer times and network behaviour, and tools to help determine application run times.

3.1 Tools to Determine Whether the Grid Is Up

Last year's workshop identified that one of the first questions applications seemed to ask was "Is the Grid up?", not only so application scientists could determine where to run their codes, but so that sites that were part of larger collaborations could show that they had met their service level agreements.

3.1.1 Inca

The **Inca** test harness and reporting framework has been developed for the automated testing, benchmarking and monitoring of Grid systems. It includes mechanisms to schedule the execution of information gathering scripts and to collect, archive, publish, and display data.

Originally developed as part of the validation and verification support for the TeraGrid project, Inca is a general framework that can be adapted and used by other Grids. As a result, Inca supports a diverse set of use cases, including service reliability, monitoring, benchmarking, site interoperability certification, and software stack validation.

Using the Inca framework, one can identify whether a Grid supports the necessary resources, Grid services, data, and certification to allow an application to run. In this sense it is able to tell a user *whether the Grid is up*, albeit with regard to a specific application and associated data set. This deployment testing not only ensures a continuous environment of Grid services and resources but also extends analysis to when an application is installed, executing, and accessible to users. In addition, one can monitor an application to discover whether it is providing an acceptable level of performance for the end user. As well as user-specific reporting, Inca can be configured to provide Grid/virtual organisation management (which allows user-level problems to be detected and fixed before a user notices a change in service); see Figure 1.

An important component of the Inca reporters is the *capturing of the context of execution*. This is seen as key to repeatability, allowing such information as what commands were run, at what time, on what machine, with what inputs, and delivering what result.

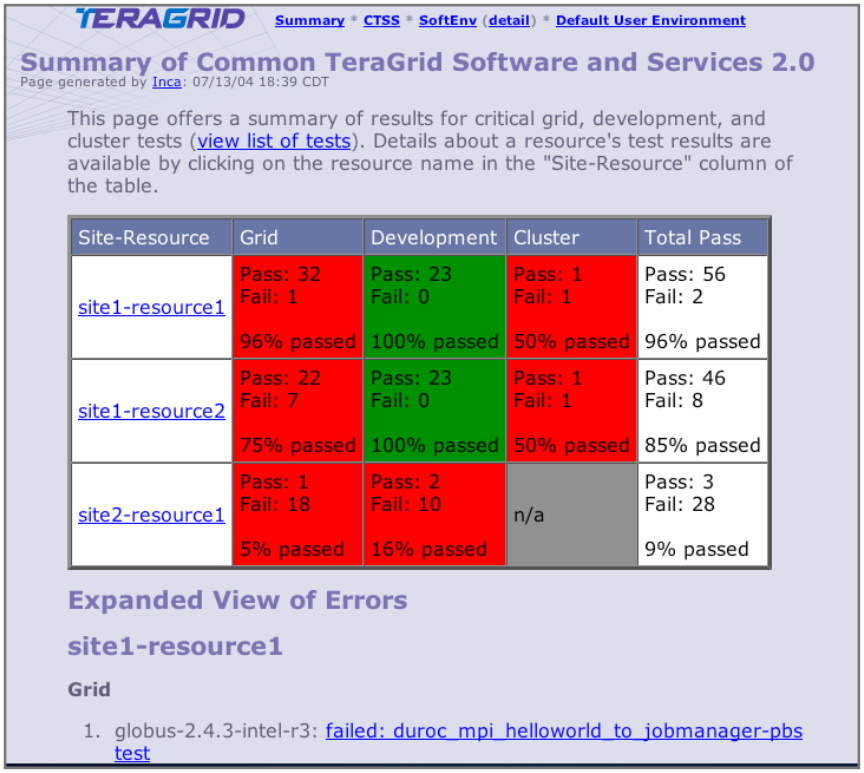

Figure 1: Example of summary status information generated by Inca on the TeraGrid.

The Inca project is funded by the San Diego Supercomputer Center (SDSC) and the National Science Foundation (NSF). For more information on the Inca project, see http://inca.sdsc.edu/. Inca 2.0 is to be released mid- to late 2005 and will include additional features for improved storage and archiving capabilities, scalability, usability, performance, security, and Condor integration.

3.1.2 Hawkeye

The **Hawkeye** tool utilises the technologies already present in Condor and ClassAds to provide rich mechanisms for collecting, storing, and using information about resources. Hawkeye can be used to monitor various system attributes (including system load, I/O, usage, and runaway processes) and therefore can be used as a basis for system management (e.g., monitoring the health of a resource pool or monitoring the health of a particular Grid site).

Since Hawkeye is based on the Condor system, it allows flexible configuration and is therefore easily customised. Hawkeye works by configuring Condor so that it periodically executes specific programs (typically scripts). The output has the form of ClassAds/value pairs, which are themselves added (using defined naming conventions) to the machine ClassAd. ClassAds can then be displayed by using `condor_status`.

Hawkeye can be found as part of the standard Condor installation and is typically used to alert the systems administrator when something goes wrong (for example, the identification of low disk space on a server or a CVS lock that has been held for more than 20 minutes). Visualisation tools are provided to aid analysis.

Hawkeye is one of the main tools used to monitor the 1400-node Condor pool at the University of Wisconsin. More information regarding the Hawkeye tool and its capabilities can be found at http://www.cs.wisc.edu/condor/hawkeye/.

3.1.3 LCG

LCG currently employs four forms of monitoring: reporting on the status of a given site; reporting on what resources are currently being used; accounting, that is, how many resources have been allocated to a particular virtual organisation; and EGEE quality assurance. These monitoring activities are not necessarily well connected, and a number of disparate tools have been developed to support these monitoring activities.

In the case of site status, a series of site functional tests are run automatically at every site. These tests may involve simply asking questions of systems or may involve running test jobs. These tests are defined as critical and noncritical. If a site consistently fails a critical test, then automated messages will be sent to the site, and eventually it will be removed from the system if the error persists. Site status information is widely published (and can be found at http://goc.grid.sinica.edu.tw/ and http://goc.grid-support.ac.uk/gridsite/gocmain/). Higher-level mapping information is also available; see Figure 2 for an example. **GridIce** is the main tool used for supporting this site monitoring activity.

Accounting is based on the records of local batch systems. This information is logged and published by using R-GMA. Quality assurance is based on overall job success and job throughput (for a given time period), and this information too is widely published and is used during software upgrading across multiple sites.

Monitoring forms the backbone of the LCG *Service Challenge* activities—tightly controlled preparation, setup, and service activities, designed to ensure delivery by April 2007 when the LHC service is finally commissioned.

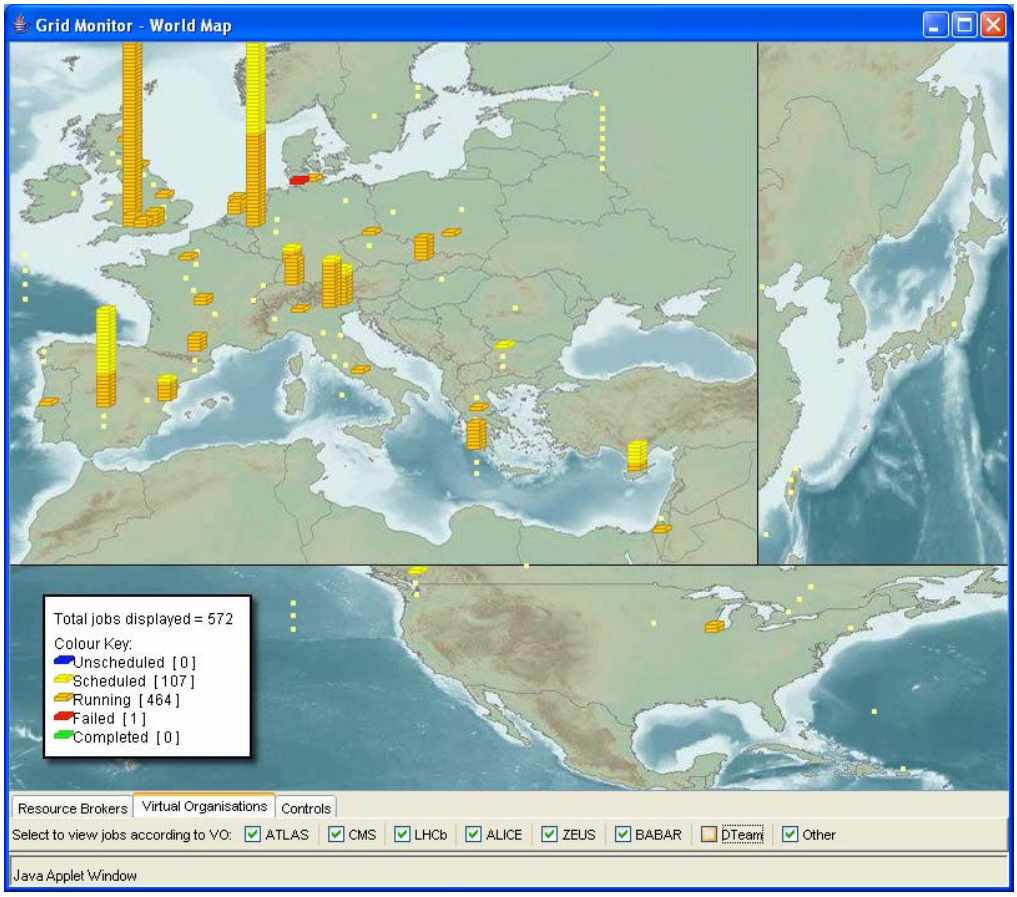

Figure 2: Global-view of the LCG site monitor.

3.2 Tools to Understand job/File Transfer Failures

3.2.1 NetLogger

NetLogger (*Net*worked Application *Logger*) is a methodology and supporting toolset for monitoring, in practical settings, the behaviour of all the elements of the application-application communication path in order to determine exactly what is happening within a complex system.

In NetLogger, distributed application components, as well as some of the operating system components, are modified to perform precision time-stamping and logging of *interesting* events at every critical point in the distributed system. The events are then correlated with the system's behaviour in order to characterise the performance of all aspects of the system and network during actual operation. The monitoring is designed to facilitate the identification of bottlenecks, performance tuning, and network performance research. The tool also allows accurate measurement of throughput and latency characteristics for application codes; as such it includes a tool for analysing monitoring events based on visualisation of the timestamp correlated event data (see Figure 3).

Since the reporting of information from large installations can be difficult to interpret, NetLogger has an automatic anomaly detection tool,called `nlfindmissing`. Additional management tools are provided for the grouping (and compression) of log data, with `nldemux` as an example.

.

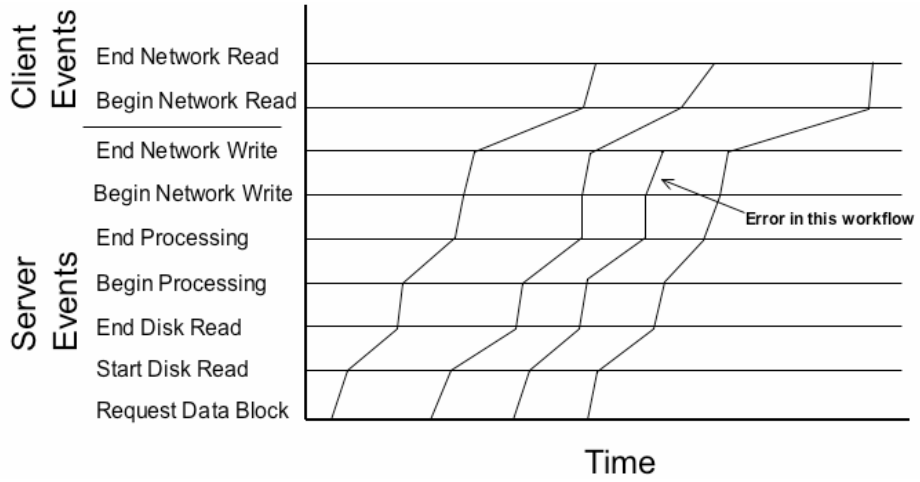

Figure 3: Analysing NetLogger monitoring events based on visualisation of the timestamp-correlated event data.

More information on NetLogger can be found at http://dsd.lbl.gov/NetLogger/.

3.2.2 PerCo

Job/file transfer failure is supported on RealityGrid through the **PerCo** (Performance Control) system. PerCo offers two-tiered hierarchical performance control, comprising a Component Performance Steerer (CPS), which wraps each application component and aims to maximise the performance on the deployed platform, and an Application Performance Steerer (APS), which deploys components on available resources and maximises the performance on the allocated platforms. PerCo operates by sitting on top of any existing external resource allocation system.

In PerCo, components progress via a sequence of *progress points*, at each of which a component calls out to its CPS for any component-specific performance control actions (local actuation that requires components to be *malleable*). Certain progress points are also *safe-points* (i.e., the component is in a state that permits it to be redeployed); and, at these points, the CPS can call out to the APS for redeployment-based performance control actions (the APS means of actuation).

An advantage of this technique is that it lends itself to those collaborative projects that require application scientists to link multiple models, which themselves may have well-defined performance control criteria and models. Performance control scenarios using PerCo have been demonstrated in the TeraGyroid experiment (conducted during the RealityGrid demonstration at SC'2003), in other applications such as IntBioSim, and in hurricane tracking.

For more information, see http://www.cs.man.ac.uk/cnc/.

3.3 Tools to Help Predict File Transfer Times and Network Behaviour

Network performance monitoring is crucial to the Grid. Measurements are required for several tasks:

- Debugging networks for efficiency, an essential step for those wishing to run data intensive applications
- Grid middleware and applications to make intelligent use of the network, optimising their performance by adapting to changing network conditions (including the ability to be "self-healing")
- Supporting the Grid "utility computing" model and ensuring that the differing network performances required by particular Grid applications are provided, via measurable SLAs

In addition, network forecasting and anomaly detection provide the basis for a number of services, including alerts for network administrators (e.g., if there are sudden changes in bandwidth), alerts for systems administrators (e.g., if the operating system or host metrics change) and the monitoring of security.

3.3.1 PingER

If the focus is on attaining high performance on a few hosts, which themselves need to send data to a small number of collaborator sites (as captured in the HEP tiered model), then there are a number of available tools for regular measurement taking, including `ping` (for round-trip-time and connectivity testing), `traceroute`, `pathchirp`, `ABwE` (for packet pair dispersion), `iperf` (both single- and multistream), `thrulay` and `bbftp` (for data and file transfer speeds). Associated with these are a number of supporting analysis and visualization tools, an example of which can be seen in Figure 4.

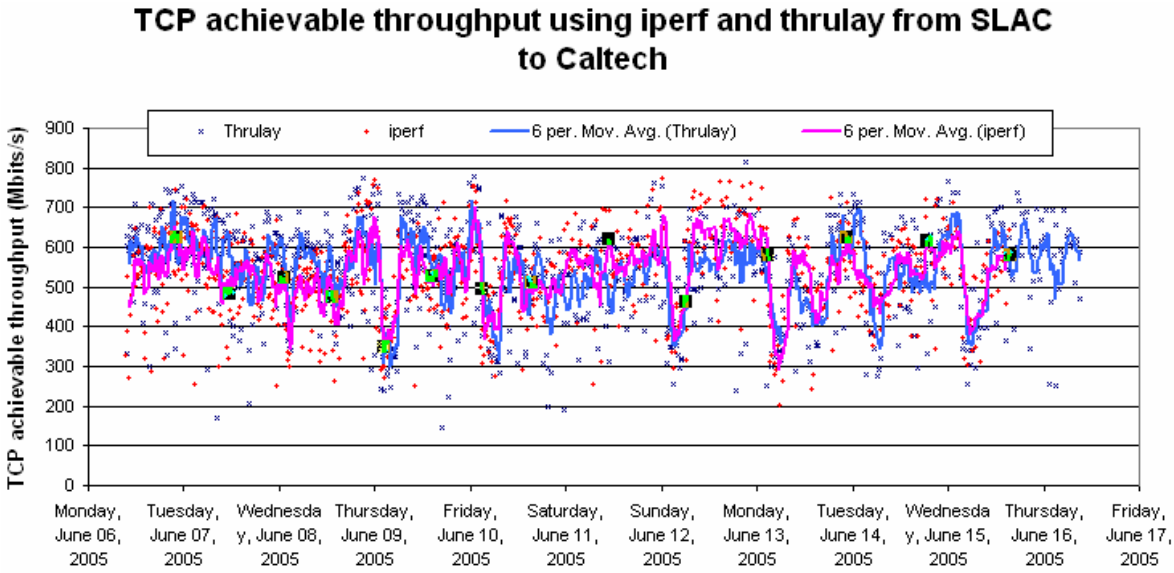

Figure 4: Sample of work undertaken at SLAC, displaying periodically monitored network performance.

3.3.2 NWS

A number of other tools measure network link bandwidth (including nws, nttcp, iperf, netperf, treno, remos). Each of these tools uses a slightly different technique, however, and therefore the results of these tools can be different. To date no clear methodology has emerged for comparing the output of these tools to see whether the results are, in fact, the same.

When evaluating tools in this area, essentially two questions arise: How can we tell whether the results produced by two tools are consistent with each other? and How can we tell whether measurements produced by the two tools convey the same amount of information? The answers to the questions are important if, for example, new tools are to be developed (and compared with those that already exist), if information is to be aggregated from more than one tool, and if techniques are to be compared (e.g., intrusive probing vs. nonintrusive probing).

Research that aims to address these issues is being undertaken at the University of California – Santa Barbara. In this research, techniques are being devised to decide whether two time series are consistent with each other, and methods of autocorrelation of differences are being developed to determine whether two time series contain the same information. This research is being undertaken in the context of the **NWS** (Network Weather Service), a distributed system that periodically monitors and dynamically forecasts the performance that various network and computational resources can deliver over a given time period.

Details on the Network Weather Service can be found at http://nws.cs.ucsb.edu.

Catalogues of network measurement tools can be found at
http://www.slac.stanford.edu/xorg/nmtf/nmtf-tools.html and
http://www.caida.org/tools.

3.4 Tools to Help Determine Application Run Times

A number of advantages can be gained by predicting the performance of Grid applications. Anticipating workload behaviour prior to run time can, for example, have a significant effect on the ability to schedule tasks in a way that provides a dependable degree of system performance. Moreover, these benefits accumulate as the size and heterogeneity of the underlying supporting architectures increase.

Approaches to performance prediction in this context can be loosely grouped into three main areas: statistical approaches (based on measurement), simulation-based approaches and modelling techniques. More recent research in this area, such as **Prophesy**, use a hybrid of techniques combining statistical curve-fitting with modelling.

3.4.1 Prophesy

The Prophesy framework consists of three major components: data collection, data analysis, and three central databases. The data collection component focuses on the automatic instrumentation of software code at the level of basic blocks, procedures, or functions. The default mode consists of instrumenting the entire code at the level of basic loops and procedures, although the granularity can be modified by the user. Resulting performance data is stored in a database and forms the basis for further analysis. Models are constructed from the monitored data, model templates, and system characteristics.

Prophesy allows the development of linear and nonlinear models and, when combined with data from the system database, can be used by the prediction engine to predict the performance on a different compute platform. These models can then be used to give insight into which machine may provide the best performance for a given implementation of a kernel and what happens when different features of the system change.

Prophesy has been applied to a number of Grid problems including the GriPhyN Grid2003 test-bed project and the Laser Interferometer Gravitational wave Observatory (LIGO) project. For more information on Prophesy, see http://prophesy.cs.tamu.edu/.

3.4.2 IXI

Similar modelling and monitoring work has been applied by the High Performance Systems Group at the University of Warwick to one of the UK Grid exemplars. The **IXI** (Information eXtraction from Images) project aims to apply e-Science research to medical imaging. The IXI infrastructure has applications in biomedical research, drug discovery, and healthcare and currently uses the UK National Grid Services as a test platform.

Analytical modelling research based on the PACE toolkit has resulted in performance prediction models with <10% error for some of the core medical imaging components. The PACE toolkit is, however, less able to handle largely data dependent code. Since the image registration of MRI or CT scans can take between 10 minutes and 24 hours (depending on the data set), some form of prediction is necessary to support clinicians (performing computer assisted surgery, for example) and research scientists (engaged in drug discovery).

Several new techniques for the performance prediction of data-dependent Grid applications are being developed through a partnership between application scientists and tool developers. More information on this work can be found at
http://www.dcs.warwick.ac.uk/research/hpsg/ and
http://www.nesc.ac.uk/action.projects/project_action.cfm?Title=121.

3.4.3 Active Harmony

The **Active Harmony** project is seeking to expand the interface between applications and system software. As a result, an API has been written that allows applications to be written to be resource-aware. In particular, the model is based on having export options to the runtime system that then select particular values for the application based on the available resources and other workload on the nodes. In addition, the applications indicate the expected resource utilisation for each option and resulting performance level (e.g., completion time or speedup). For example, a parallel application might expose an option that would select the number of nodes that the program can use. The expected resource information might indicate that the application can effectively use 16 nodes but that performance improves marginally after that. The Active Harmony system could use this information to run the job on more than 16 nodes if these were available, but only on 16 nodes if other jobs could more effectively use these nodes.

This work has been used in a number of applications, including cluster-based Web Service benchmarking and management and application to scientific codes such as POP (the Parallel Ocean Program), GS2 (a physics application developed to study low-frequency turbulence in magnetised plasma), and PETSc (middleware and data structures for scientific applications). More information on Active Harmony can be found at http://www.dyninst.org/harmony/.

4 Simulation

The large majority of research activities in the context of Grid systems and applications cannot rely on purely analytical developments but must instead rely on *experiments*. Indeed, most authors in the field perform experiments and report on results obtained on actual Grid platforms. Although results obtained with such direct experimentations are believable and demonstrate that the authors' approach (e.g., algorithms, distributed system design) can be implemented in practice, direct experimentation has a number of limitations. First, direct

experimentation can be time-intensive and labor-intensive. For instance, evaluating different resource allocation strategies for an application that runs for several hours could take many days. Also, running experiments requires that the system or application under study be fully implemented, most likely with many different implementations to compare algorithmic and design alternatives. Second, it is not clear what testbed should be used. On the one hand, one can use a small, ad hoc testbed, which is stable and highly controlled so that experiments can be performed reliably and repeatedly, but which is hardly representative of a real-world Grid platform. On the other hand, using a production Grid platform is challenging because experiments can be disruptive to the platform's users and because these users can themselves be disruptive to the experiments in nondeterministic and nonrepeatable ways.

Hence, it is difficult to obtain both realistic and statistically significant experimental results bya direct experimentation on Grid platforms. Furthermore, results obtained by direct experimentation are inherently limited to the configuration of the testbed(s) at hand, precluding the exploration of "what if" scenarios. Moreover, it is extremely difficult for researchers to attempt to reproduce (and improve on) results obtained by direct experimentation and reported by authors in research papers, which should be the basis for scientific advances.

One way to address most (and possibly all) of the shortcomings of direct experimentation is to employ *simulation*. Simulation has been used with success in several fields of computer science such as networking and microprocessor design. Simulation had also been used in the area of parallel computing research, but the models used have been simple and hardly justifiable for Grid platforms. Fortunately, a number of developments and advances in the past few years, as well as simulation efforts from the networking area, provide the first steps toward the establishment of a technology and methodology basis for simulation in the area of Grid computing research.

The first step in running Grid simulations is to generate synthetic Grid platforms, including two main elements—the network resources and the compute resources—with, in both cases, models for "background" resource utilisation and availability.

The network resources – One distinguishing features of Grid platforms when compared to traditional parallel computing platform is the complexity of the network. This complexity leads to heterogeneity of bandwidths and of round-trip times as well as complex bandwidth sharing among competing network connections. Modelling the network topology is key for simulating these behaviours, and the network community has developed several generations (random, structural, and degree-based) of tools to generate graphs that are representative of the topology of the Internet. It is now accepted that degree-based generators, that is, ones that respect the power laws observed in real networks, should be used to generate large-scale topologies such as the ones seen in large-scale Grids, while structural generators should be used to generate small-scale topologies. A difficult issue is that of annotating the network links in the generated topology with representative characteristics (bandwidth, latencies). One approach consists in annotating each link individually with physical characteristics based on simple rules, and then modelling possible background traffic explicitly as random traffic between many random end-points. Another approach consists in annotating network links based on observations of actual end-to-end network connections. In the former, it is not clear how to annotate links and generate background traffic in a way that is representative, while in the latter it is not clear how to annotate links based on end-to-end information.

The compute resources – Modelling representative Grid resources is still in early stages. Two projects in the past two years have provided first steps. The GridG project let by Dinda provides a full framework for generating synthetic Grids, including compute resources. Also, the work by Kee et al. at SC'04 has developed statistical models based on a sample of hundreds of commodity clusters, which can be used to generate sets of representative (current

or future) cluster configurations. Such synthetic clusters can then be attached to the end-points of a synthetic network topology to obtain a full Grid configuration. One important issue here again is to model resource availability. This can be done via traces of resource usage (many trace datasets are available for batch-scheduled resources and for desktop resources), or via actual models of resource usage (e.g., ones developed for batch-scheduled resources by Feitelson et al.).

Once a synthetic Grid configuration has been instantiated, one must then execute a simulation. Simulation techniques differ in the level of simulation abstraction, ranging from "mathematical simulation" to "emulation". While a number of projects in the past few years have implemented Grid simulations, two projects have been more prominent in the community: the MicroGrid and the SimGrid projects. MicroGrid lies on the emulation side of simulation, with execution of actual Grid middleware and application code on Grid resources that are virtualised on physical resources and with packet-level simulations of network communications. At the other extreme, SimGrid uses mathematical models of resource sharing to simulation application execution without execution of application code but rather based on a specification of the application's resource requirements. In the former approach simulation time can be much larger than simulated time, while in the latter approach simulation time is dramatically shorter than simulated time. However, it is expected that simulation accuracy in the former is higher than in the latter. But this is an open question, and studies in other fields such as computer architecture indicate that the trade-off between simulation speed and simulation accuracy is unclear and that less detailed and thus faster simulations can be more accurate.

The key question for simulation, and therefore for Grid simulation, is that of validation: How accurate and close to the real world is simulation? Validation is known to be difficult, and much work needs to be done in the area of Grid simulation. Ongoing work as part of the GRAS project is attempting to provide a framework to systematically compare abstract simulation, emulation, and the real world. Necessary steps for the advancement of Grid simulation, and consequently the advancement of Grid computing as a scientific discipline, are the establishment of an accepted set of tools and methodologies for Grid simulation as well as of a repository of synthetic Grid configurations and measurement datasets that can be used by the community to instantiate representative simulations. Only with these will Grid researchers be able to easily reproduce and improve on each other's results.

5 State of Play

As a result of the open discussion at this workshop it was agreed that there is generally a ***mismatch between application scientists and tool developers*** in this research area. It is not clear to the tool developers what the application scientists want and how much they are willing to reengineer their applications to use available performance services. There is also a tendency by the tool developers to address highly complex issues, but in many cases the solutions required by the application scientists are simpler than that (including good reliable file transfer and a global Grid `top` command). In many cases, complex issues are addressed because the tool developers are funded to do novel or research-oriented work, as opposed to hardening code or simple engineering solutions.

It was agreed that there is a clear ***difference between short and long-term needs***, and areas such as resource brokering, replica location, and metacatalogues, while clearly necessary, remain secondary (future) requirements for application scientists. Therefore, a disconnect exists between the tool research discussed in papers and its eventual application to middleware, applications, and users. It is also recognised that moving from papers to implementation to the final product (robust, user friendly, installable toolkits) is difficult.

Indeed, one could argue that this is as much engineering as it is *new science*, and for this reason there is a disproportionately small amount of funding for this last step.

One interesting problem that the tool builders had not realised they faced was that ***performance simply isn't on the critical path for many application projects***. Most projects are still in the phase of getting their code executing correctly on the distributed resources and growing their infrastructure. It was seen as hard enough to get things simply running; whether they were using resources efficiently or achieving good performance was secondary for many projects. As an example, a large number of additional application scientists were contacted to attend this workshop, travel expenses covered, but chose not to.

There is also a clear ***need for reliable solutions,*** and the application scientists emphasised that reliable solutions are as important as, if not more important than, high-performance solutions. Performance and reliability are complementary sets of needs, and short-term research solutions should aim to address both these concerns.

In order to direct future research (and funding), it was clear that ***a needs analysis is required***. This should clarify the needs of the application scientists and focus the tool developers on what is needed with respect to the tools. In addition, there should be a closer interaction between the tool developers and application scientists. It is not clear, for example, whether the application scientists are aware of what tools and techniques are available to assist them with performance analysis. The communication required is therefore very much two-way. There are a number of possible ways forward:

- Encourage more collaborative (application scientist, tool developer) projects, to both identify areas of concern and also promote pragmatic tools development.
- Promote exemplars of good practice (tools and applications people working together), as this will encourage more of the same.
- Provide clear sources of information, since at present it is not clear where to seek information about performance tools research.

It is not clear that the Global Grid Forum, in its current form, is the right venue in which to pursue these issues. and A more targeted working group involving application scientists and tool developers is probably more appropriate. A number of recommendations and follow-up initiatives were made (Section 6).

There is a concern that by not integrating the performance-tool developer and application scientist, two situations will arise: the tool developer will continue to engineer increasingly complex solutions that are difficult to apply in practice, and the application scientists will develop their own point solutions for particular applications, which are not transferable to other applications, are not scalable, or are not as efficient as a tool domain expert might be able to develop. Through collaborative applications/tools projects, the community should strive for ***performance tools that offer practical, robust solutions***, yet allow the generic research developed during this work to be clearly visible, reproducible, and scalable. There should also be continued effort to compare tools, and perhaps this is something that can be addressed, for example, through simulation.

6 Recommendations of the Meeting

As a result of the meeting, several possible avenues for future work in the area of Grid and performance were identified, although the current lack of funding for these projects may limit the timeliness of their accomplishment.

1. Survey of application requirements

A survey of the requirements of the application scientists is needed. Several suggestions were made for format (Web form, paper survey, in-person meetings), and in general it was agreed that face-to-face discussions would be best, similar to what was done last July for OMII and Globus Toolkit requirements gathering (http://www-unix.mcs.anl.gov/~schopf/Pubs/25uk.tr.pdf). With current funding and time constraints, however, large group meetings were also suggested (see point 6 below).

2. Catalogue of available tools

A catalogue of available performance tools should be compiled. This should encapsulate existing catalogues (e.g., http://www.slac.stanford.edu/xorg/nmtf/nmtf-tools.html, http://www.caida.org/tools, and the APART project's compilation of performance tools for the Grid http://www.lpds.sztaki.hu/~zsnemeth/apart/repository/gridtools.pdf).

More useful than a simple catalogue would be something similar to a Wiki with a rating system. Many of the application scientists expressed the view that there were too many tools and it was hard to know which were production-ready, but a site that allowed feedback would be extremely useful. It was suggested that this catalogue should provide some means of *rating*, so that users of these tools could provide feedback on their effectiveness and user experiences

3. Identification of key areas in Grid performance research

Based on points 1 and 2, short-term and long-term goals of this community should be defined, and these recommendations should be made available to the appropriate funding bodies.

4. Joint application scientist/tool developer projects

Encouraging joint application scientist/tool developer projects is considered essential. Efforts should therefore be made to promote this research, including a journal special issue that would highlight success stories, and features in newsletters (NeSC's newsletter, the SDSC newsletter, ScienceGrid, etc.).

5. Identification of funding

Discussions will be held with funding bodies to determine how best to support multicountry and application scientist/tool developer projects.

6. Community outreach

Further discussion with, and between, tool developers and application scientists is needed. One suggestion was to hold a BOF at a major HPC conference such as Supercomputing or the UK eScience All Hands Meeting.

7 Future IGPW Meetings

Discussions were held as to whether an IGPW meeting should be arranged for next year. The feeling of the meeting was that IGPW is a productive forum and that in order to preserve

continuity, it would be good to have a meeting in 2006. A poll of previous attendees will be taken in six months time to ensure that this is still the general feeling.

The theme of the next workshop would be that of *integration* between application scientists and tool developers. More application scientists would be encouraged to attend the workshop, and paired talks would be arranged, highlighting case studies where tool developers have worked alongside application scientists to meet specific performance needs. Demonstrations of tools and applications would be strongly encouraged.

Since the previous two meetings have been held in the UK, it is likely that the next meeting would be elsewhere (most likely in the United States). An appropriate venue would be selected in order to maximise participation of research scientists in this field.

The JISC/NSF funding for this meeting has now expired, and therefore further funding will need to be sought.

Acknowledgements

This work was supported in part by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract W-31-109-ENG-38, and by NSFAward #0432288, as well as support from the Joint Information Systems Committee (JISC) and Microsoft. We also acknowledge the support of the National e-Science Centre in the UK for their assistance with the local arrangement for this year's workshop.

Appendix A: Agenda and List of Participants

7.1.1 June 22

9:00-9:30	Registration
9:30-10:00	Introduction and Overview Jennifer Schopf, Welcome
10:00-12:00	Applications: Three talks by application scientists, focusing on performance Andrew Porter, RealityGrid Justin Burruss, FusionGrid Birger Koblitz, LCG2
12:00-12:30	Applications Panel: An opportunity for tool developers to ask questions of a panel of application scientists
12:30-1:15	Lunch
1:15-2:45	Themed Tool Talk(s): Is the Grid up? Shava Smallen, Inca Nick LeRoy, Hawkeye David Colling, LCG
2:45-3:15	Coffee break
3:15-4:15	Themed Tool Talk(s): Why did my job/file transfer fail? Brian Tierney, NetLogger John Gurd, PerCo
4:15-5:30	Tool Panel: An opportunity for application scientists to ask questions of a panel of tool developers Start of collection of issues for final panel to address.

7.1.2 June 23

9:00-10:00	KEYNOTE Simulation - its role with applications and Grid performance Henri Casanova
10:00-10:30	Coffee Break
10:30-11:30	Themed Tool Talk(s): How long will my file transfer take? Matthew Allen, NWS Les Cotrell, PingER
11:30-1:00	Themed Tool Talk(s): How long will my job take to run? Seung-Hye Jang, Prophesy Daniel Rueckert, IXI I-hsin Chung, Active Harmony
1:00-2:00	Lunch
2:00-3:30	Breakout sessions on specific performance topics: Are there any disadvantages to the approaches used by the tools for measuring/prediciting performance? If so, which of those issues do the applications people need addressing, and what is the best way forward
3:30-4:00	Workshop Summary Panel to summarise results, understand how to expand this work to the general case

Mr Ahmed Abdelrahim, National e-Science Centre
Mr Ali Afzal, Imperial College London
Mr Matthew Allen, University of California - Santa Barbara
Dr Anne Benoit, University of Edinburgh
Dr Richard Blake, CCLRC - Daresbury Laboratory
Mr Justin Burruss, General Atomics
Dr Henri Casanova, University of California - San Diego
Dr I-Hsin Chung, IBM Thomas J. Watson Research Center
Dr. David Colling, Imperial College
Dr Les Cottrell, Stanford University
Dr Karim Djemame, University of Leeds
Dr Jack Dongarra, University of Tennessee
Mr Brian Foley, University of Warwick
Mr Stephen Gilmore, University of Edinburgh
Prof John Gurd, University of Manchester
Mr Jesus Israel Hernandez, University of Edinburgh
Prof Tony Hey, EPSRC
Dr Seung-Hye Jang, Texas A&M University
Dr Stephen Jarvis, University of Warwick
Dr Tahar Kechadi, University College Dublin
Dr Birger Koblitz, CERN
Mr Mark Leese, CCLRC-Daresbury Laboratory
Mr Nick LeRoy, University of Wisconsin
Miss Lisha Ma, Heriot-Watt University
Dr Andrew Steven McGough, Imperial College London
Dr Steven Pickering, University of Leeds
Dr Andrew Porter, University of Manchester
Mr Graham Riley, University of Manchester
Dr Daniel Rueckert, Imperial College London
Dr Jennifer Schopf, Argonne National Laboratory and NeSC
Mr Andrey Shevel, State University of New York
Ms Shava Smallen, San Diego Supercomputer Center
Dr Nigel Thomas, University of Newcastle
Mr Brian Tierney, Lawrence Berkeley National Laboratory

Appendix B: Abstracts (in agenda order)

Performance Considerations within RealityGrid

Andrew Porter, University of Manchester, RealityGrid

The focus of the RealityGrid project is the use of the Grid to facilitate the simulation of condensed-matter systems such as material surfaces, miscible fluids, and macro-molecules. The scientists doing such work typically have existing codes (written in a variety of languages) for doing the calculations and require access to powerful, parallel computing resources. The RealityGrid project has provided a set of tools for use by the application scientists. These tools include functionality for launching calculations on GT2-based Grids, monitoring and interacting with a running job (including the provision of on-line visualisation) and managing the checkpoints produced by a job or set of jobs. The latter functionality has been used to aid tasks such as parameter space exploration and job migration. In this talk I will discuss the aspects of the project where performance is critical to the application scientists, the level of functionality currently offered and what additional features we would consider desirable.

Grid Performance and Fusion Science

Justin Burruss, General Atomics, FusionGrid

Fusion research seeks the development of an environmentally and economically attractive power plant. Fusion science is advanced through experiments carried out on fusion devices called “tokamaks,” such as the DIII-D tokamak in San Diego, California. Fusion experiments are interactive—not “batch mode”—and require rapid data analysis. Furthermore, because fusion researchers are geographically dispersed, fusion experiments require scientists to work remotely, both to lead and otherwise participate in experiments. For these reasons, managing data analysis throughput and remote collaboration reliability are the chief performance concerns. As fusion data analysis moves off of the local network and onto distributed resources on the WAN, performance problems become trickier to diagnose and repair. Solutions to Grid performance management are required.

Catalogue Access on the Grid
Birger Koblitz, CERN, LCG2

I present the studies of the ARDA project on the access to metadata and file catalogues. Both services rely on databases as a backend, and a fast access is very important to user applications, while security needs to be taken into account as well as to enable the user to make use of the capabilities of a database. I will present performance comparisons on the LFC and Fireman EGEE file-catalogues as well as the ARDA metadata service, which allows one to directly compare SOAP and text-streaming access to a database. Finally, some design considerations of the ARDA/gLite metadata interface are presented.

Inca Test Harness and Reporting Framework
Shava Smallen, San Diego Supercomputer Center, Inca

Running applications on the Grid remains challenging for users. This difficulty is in part due to knowing when Grid services and resources are up and available and possible inconsistencies in the available Grid software environment. Inca is a flexible framework for the automated testing, benchmarking, and monitoring of Grid systems. It includes mechanisms to schedule the execution of information gathering scripts and to collect, archive, publish, and display the results. This talk covers how Inca can be utilised to detect installation, configuration, user environment, and/or performance problems on Grids and how this information benefits users. I will also discuss how Inca is currently being used by the TeraGrid project, a NSF-funded Grid effort composed of nine sites across the United States. Finally, I'll conclude with a status summary of our next version of Inca.

Keeping a Hawkeye on the Grid
Nick LeRoy, University of Wisconsin, Hawkeye

With the increasing availability of Grid computing technologies, we provide researchers with unprecedented access to computing resources. As the size of individual clusters and the number of these clusters that constitute these Grids continue to grow, so do the associated maintenance costs. Advanced and automated tools are increasingly required to keep these resources functioning properly. In this talk I will discuss Hawkeye and how it can be used to monitor various aspects of computing Grids.

Performance and the LCG
David Colling, Imperial College London, LCG

The LCG is an operational Grid currently running at 136 sites in 36 countries, offering its users access to nearly 14,000 CPUs and approximately 8 PB of storage. Monitoring the state and performance of such a system is challenging but vital to successful operation. The standard monitoring tools are described and an analysis of the performance of this LCG presented.

Techniques for Monitoring Large Loosely Coupled Cluster Jobs
Brian Tierney, Lawrence Berkeley National Laboratory, NetLogger

Science and commerce have both experienced an explosion in the sheer amount of data that must be analysed. More and more compute clusters are being used for analysing these huge data sets. However, debugging and tuning clusters require specialised tools. Current cluster performance tools are more oriented towards tightly coupled parallel applications. We describe how the NetLogger Toolkit methodology is more appropriate for loosely coupled cluster computing, and we describe our new automatic workflow anomaly detection component. We also describe how this methodology is being used in the Nearby Supernova Factory (SNfactory) project at Lawrence Berkeley National Laboratory.

Use Case Scenarios for Performance Control of Grid-based Metacomputing
John Gurd, University of Manchester

A Grid-based performance control system, PerCo, has been developed as part of the “deep track” activity in the RealityGrid project. PerCo is capable of controlling the behaviour of component-based Grid applications at two levels. Behaviour in a single component can be altered by changing actuating parameters that affect the way that the component executes in its allocated environment; this is controlled by a component performance steerer (CPS) to which the component code is attached. Behaviour of the application as a whole can be altered by moving components from one execution environment to another; this is controlled by a central application performance steerer (APS). The PerCo infrastructure is fully implemented, and is capable of cleanly closing down components, transferring them to a new platform, and restarting them correctly in the new environment. The question addressed by this presentation is, What can the PerCo system do that will improve the situation faced by a scientist using high-end computing to pursue some scientific objective by means of large-scale simulation? We use our contacts with the scientists in the RealityGrid (EPSRC) and IntBioSim (BBSRC) e-Science projects to develop practically useful use case scenarios that can help them to progress more rapidly with their scientific investigations.

Simulation for Grid Computing
Henri Casanova, University of California – San Diego, SimGrid

In this presentation we discuss the need for well-established simulation practice and technologies for the purpose of conducting research in the area of Grid computing. After identifying the need for Grid simulation and placing today’s state of the art in perspective with that in other areas of computer science, we discuss two main issues: the generation of synthetic Grid platforms for simulation purposes, and the simulation of applications on these platforms. We review today's main approaches and discuss the advantages and shortcomings of current technology. We discuss several open questions, such as those regarding the trade-offs between simulation speed and simulation accuracy, and we conclude with a description of recent efforts that attempt to answer these open questions.

Comparing Performance Measurement Time Series
Matthew Allen, University of California - Santa Barbara, NWS

Testing link conditions is standard practice for active Internet users. Network administrators monitor link performance to detect disruptive behaviour, determine performance bottlenecks, and locate faulty routers. Wide-area distributed applications developers use measurements to predict performance or minimise communication time between nodes. To this end, a large number of tools have been developed to measure link bandwidth. Iperf, netperf, NTTCP, and the Network Weather Service (NWS) all measure network performance through active network probing. These techniques, however, all differ slightly in their methodology, causing them to report different results. In this talk, we attempt to understand whether these different methodologies affect the measurements produced. In particular, we try to develop techniques to compare measurements produced by different tools.

Forecasting Network Performance
Les Cottrell, Stanford University, PingER

Predicting how long a file transfer will take requires forecasting network and application performance. However, such forecasting is beset with problems. These include seasonal (e.g., diurnal) variations in the measurements, the increasing difficulty of making accurate active low network intrusiveness measurements especially on high-speed (>1 Gbit/s) networks and with Network Interface Card (NIC) offloading, the intrusiveness of making more realistic active measurements on the network, the differences in network and large file transfer performance, and the difficulty of getting sufficient relevant passive measurements to enable forecasting. We will discuss each of these problems, compare and contrast the effectiveness of various solutions, look at how some of the methods may be combined, and identify practical ways to move forward.

Performance-Directed Resource Allocation
Seung-Hye Jang, Texas A&M University, Prophesy

Grid systems provide vast compute and data resources to users for large-scale applications such as cosmology, ocean modelling, and gravitational-wave physics. One of the major issues to be addressed with Grids is that of resource selection. In this talk, we will quantify the advantages of using performance prediction versus load information for resource selection. Our work uses the Prophesy infrastructure to predict application performance on different sites. Prophesy uses historical data to generate analytical performance models for predictions. The quantification is based upon two case studies. The first case study involves a large-scale scientific application, called GEO LIGO, for which the experimental results indicate an average of 33% performance improvement as compared to a load-based method. The second case study involves a Web-based, educational application, called AADMLSS, for which the results indicate an average of 10% performance improvement as compared to a load-based method.

Performance in Medical Image Computing
Daniel Rueckert, Imperial College London, IXI

In this talk we will describe why performance modelling and predication are crucial for many applications in medical image computing. We will focus on Grid applications using image registration, which are employed in computer-aided diagnosis and computer-assisted surgery.

Towards Automatic Performance Tuning

I-hsin Chung, IBM Thomas J. Watson Research Center, Active Harmony

Software today makes extensive use of libraries and reusable components in order to speed development. However, libraries used by an application may not be performance tuned to the application's need. To address this issue, we developed the Active Harmony automated runtime tuning system. I will describe the interface used by programs to make applications tuneable. I will also present the optimisation algorithm used to adjust application parameters together with a library to expose multiple variations of the same API using different algorithms. In order to speed the tuning process, our tuning server utilises historical data. When the system to be tuned has numerous parameters, our system uses techniques to prioritise parameters and identify the relations among parameters to avoid trying unnecessary configurations. In addition, for homogeneous compute nodes, we use parameters replication to tune all nodes as one. We have successfully applied the Active Harmony system to commercial and scientific applications. The experimental results show that frequently no single configuration performs well for all kinds of workloads. The performance improvement cannot easily be achieved by tuning individual components for such a system. For an e-commerce Web site, Active Harmony helps the system adapt to different workloads and improve the performance up to 16% (70% with reconfiguring node roles); and for a parallel plasma simulation code, execution time is reduced up to 70%.